

AI Based Fake News Detection Using NLP

Santhosh N

Master of Computer Applications, Nehru Institute of Information Technology and Management, Tamil Nadu, India
Reg. No: 721524622044 | Batch: 2024–2026 – MENTOR – Mr. P. Sivakumar, Asst. Professor [Dept. of Computer Applications]

Abstract—The proliferation of social media and digital news platforms has made the rapid, uncontrolled spread of fabricated and misleading content — commonly termed fake news — a pressing societal challenge, capable of manipulating public opinion, influencing elections, and eroding trust in legitimate journalism. Manual fact-checking, while accurate, is inherently too slow to match the pace of digital content distribution. This paper presents an AI-based Fake News Detection system that applies Natural Language Processing (NLP) and Machine Learning (ML) to classify news articles as Real or Fake in real time. Raw text is normalized through a fixed pre-processing pipeline — cleaning, lowercasing, tokenization, stop-word removal, and stemming/lemmatization — before being converted into numerical feature representations using Bag-of-Words, TF-IDF, Word2Vec/GloVe embeddings, and BERT-based contextual embeddings. These features are passed to a suite of interchangeable classifiers, including Naive Bayes, Support Vector Machine, Logistic Regression, Passive-Aggressive, Random Forest, LSTM networks, and transformer-based BERT models, allowing an explicit trade-off between accuracy and computational cost. A coefficient-driven explainability layer highlights the specific keywords and sentences that most influenced a given verdict, and all predictions are logged to a relational database for auditing and analytics. The system is delivered as a full-stack web application — a React front end and a FastAPI back end — and was validated through a layered testing strategy spanning unit, integration, functional, system, white-box, black-box and acceptance testing, with fifteen functional test cases achieving a 100% pass rate. Benchmarking of the classical classifiers indicates that Naive Bayes achieves an accuracy of 88%, an F1-score of 88.3%, and a precision of 85%, confirming that a computationally lightweight model remains competitive while transformer-based models offer superior contextual accuracy at higher computational cost. The results confirm that a scalable, explainable, and freely deployable fake-news screening tool can materially reduce the gap between the speed of misinformation and the speed of verification.

Index Terms—Fake news detection, Natural Language Processing, machine learning, TF-IDF, word embeddings, BERT, LSTM, explainable AI, text classification, misinformation.

I. INTRODUCTION

News consumption has shifted decisively from traditional print and broadcast media to online platforms and social networks. While this shift has made information more accessible and immediate, it has simultaneously opened the door to the rapid, uncontrolled spread of fake news — fabricated, misleading, or out-of-context content presented as legitimate journalism. Fabricated stories are frequently engineered to provoke strong emotional reactions, which makes them disproportionately likely to be shared before any correction can take hold, widening the gap between the speed of misinformation and the speed of manual correction [1][3].

Existing fact-checking platforms such as PolitiFact, Snopes, and Boom Live rely on human moderators and crowd-sourced reporting, a workflow that has changed little over the past decade despite the exponential growth in the volume of digital content requiring review. Automated signals used by social platforms — reduced distribution for known low-quality domains or post-hoc warning labels — remain largely reactive, applied only after a story has already begun to spread [10][11].

This paper presents an AI Based Fake News Detection system that: (1) applies a fixed, reproducible NLP pre-processing pipeline to both training and live inference data; (2) benchmarks classical machine learning classifiers against deep learning and transformer-based models to make the accuracy–cost trade-off explicit; (3) incorporates a coefficient-based explainability layer that surfaces the specific words and sentences driving a prediction; and (4) is delivered as a complete, database-backed web application capable of real-time classification and historical auditing.

II. RELATED WORK

A. NLP-Based Fake News Detection

Oshikawa et al. [3] surveyed NLP approaches to fake news detection, establishing that linguistic, stylistic and syntactic features are strong discriminators between fabricated and genuine reporting, and that generalization to unseen phrasing remains the central open challenge. Zhou and Zafarani [11] provided a fundamental theoretical framework distinguishing knowledge-based, style-based, propagation-based and source-based detection strategies, of which this project adopts a style- and knowledge-based hybrid centered on the article text itself.

B. Machine Learning and Deep Learning Classifiers

Shu et al. [10] framed fake news detection as a data-mining problem and catalogued the classical feature-engineering and classification pipeline that underlies Naive Bayes, SVM and Logistic Regression approaches. More recent comprehensive reviews [4][7] confirm that transformer-based architectures such as BERT consistently outperform classical bag-of-words classifiers on held-out benchmarks, at the cost of significantly higher training and inference resource requirements — a trade-off directly reflected in the multi-model design adopted here.

C. Stance and Survey-Based Approaches

Survey work on stance classification [5][8] demonstrates that determining how a headline relates to its body text is a valuable auxiliary signal for credibility assessment, while broader surveys of ML/NLP approaches to social-media misinformation [6][9] motivate the layered pre-processing and multi-classifier benchmarking strategy used in this system.

III. SYSTEM ANALYSIS

A. Existing System Limitations

Current verification relies primarily on manual fact-checking organizations and, secondarily, on rudimentary automated filters such as keyword matching and domain blacklists. Both approaches are bound by the pace of human research or by easily circumvented static rules, and neither provides ordinary readers with an immediate, evidence-based credibility signal at the point of reading. Table I contrasts the existing approach with the proposed system.

TABLE I. Comparative Analysis: Existing Systems vs. Proposed

Feature	Existing System	Proposed System
Verification Method	Manual review by fact-checkers	Automated NLP/ML classification
Turnaround Time	Hours to days	Real time (seconds)
Scalability	Limited by human capacity	Effectively unlimited, parallelizable
Cost	High (trained investigators)	Low – open-source stack
Explainability	Manual, written rebuttal	Automated keyword/sentence highlighting
Detection Basis	Domain blacklist / user reports	Linguistic & contextual patterns

System

B. Problem Statement

The problem addressed is the absence of a freely accessible, automated system that can classify the credibility of a news article from its text alone, in real time, while providing an interpretable justification for its verdict and maintaining a durable, queryable history of past predictions for auditing and model improvement.

IV. SYSTEM ARCHITECTURE

The system follows a layered, cloud-deployable architecture comprising five logical layers, as summarized below.

Presentation Layer: A React front end renders the article-submission form, prediction result panel, history table, and administrator dashboard, communicating exclusively through a REST API so that back-end model upgrades never require a front-end rebuild.

Processing Layer: Input validation, text pre-processing (cleaning, tokenization, stop-word removal, lemmatization), feature extraction (TF-IDF/embeddings) and classification are executed as discrete, swappable stages.

Data Access / Machine Learning Layer: Dataset collection and labeling, model training, evaluation (accuracy, precision, recall, F1-score) and a versioned model repository allow administrators to retrain and redeploy models without code changes.

Support Layer: Cross-cutting concerns — authentication, activity logging, exception handling, and data security — are centralized so that individual feature modules remain lightweight.

The end-to-end data flow follows six stages: (1) the user submits article text or a URL; (2) the Preprocessing Module cleans and normalizes the text; (3) the Feature Extraction Module converts the text into a numerical vector; (4) the trained ML/DL classifier generates a Real/Fake label with a confidence score; (5) the result, together with a hash of the input, is persisted to the Prediction_Result table; and (6) the Result Presentation layer renders a color-coded verdict to the user.

V. DESIGN AND METHODOLOGY

A. Text Pre-processing Pipeline

Raw article text is transformed through a fixed, order-sensitive pipeline so that training data and live submissions are always processed identically: (1) cleaning — removal of HTML tags, URLs, punctuation and numerical noise; (2) lowercasing; (3) tokenization into words or sub-word units; (4) stop-word removal; and (5) stemming (Porter Stemmer) or lemmatization to reduce inflected words to a common root. An optional sixth stage, part-of-speech tagging, supports advanced stylistic features such as an unusually high density of adjectives or exclamatory constructions, both empirically associated with sensationalized content.

B. Feature Extraction

Cleaned text is converted into numerical vectors using one of four interchangeable representations of increasing sophistication:

Bag-of-Words (BoW): raw word-count vectors; simple but order-agnostic.

TF-IDF: weights each term by its frequency in a document relative to its frequency across the corpus, $TF\text{-}IDF(t,d) = TF(t,d) \times \log(N / DF(t))$, down-weighting common words while highlighting distinctive ones.

Word2Vec / GloVe: dense embeddings capturing semantic similarity between words based on co-occurrence context.

BERT Contextual Embeddings: dynamic, context-aware vectors in which the same token receives different embeddings depending on surrounding text, generated by a pretrained transformer fine-tuned for the classification task.

C. Classification Models

The fitted feature vectors are passed to one of several interchangeable classifiers — Naive Bayes, Logistic Regression, Support Vector Machine, Passive-Aggressive, Random Forest (classical, scikit-learn based), LSTM networks (sequential deep learning), and fine-tuned BERT (transformer-based) — selectable at inference time without retraining the surrounding pipeline, so that the accuracy/cost trade-off can be tuned to the deployment context.

D. Explainability (XAI) Layer

For linear models, per-token coefficients learned by the classifier are mapped back onto the vocabulary. For a submitted article, each token's coefficient contributes to both a ranked keyword-influence list and a sentence-level normalized score, computed as the sum of token coefficients in a sentence divided by the sentence's token count. Sentences scoring above a positive threshold are tagged as reinforcing a Real verdict, those below a negative threshold as reinforcing a Fake verdict, giving the user a transparent, sentence-by-sentence rationale rather than an opaque probability alone.

VI. MODULES DESCRIPTION

Module 01: Data Collection: Sources, validates and merges labeled datasets (Kaggle Fake and Real News, LIAR, FakeNewsNet), removes duplicates, and produces stratified train/validation/test splits with dataset version control for reproducible experiments.

Module 02: Pre-processing: Applies the fixed cleaning, tokenization, stop-word removal and lemmatization sequence described in Section V-A consistently to both training data and live user submissions.

Module 03: Feature Extraction: Fits BoW/TF-IDF vectorizers or embedding models exclusively on training data to prevent leakage, persists the fitted vectorizer to disk, and re-applies it unchanged at inference time.

Module 04: Model Training: Trains and cross-validates the candidate classifiers, evaluates each on accuracy, precision, recall, F1-score and confusion matrix, and serializes the selected model and vectorizer for deployment; supports scheduled retraining as new labeled data arrives.

Module 05: Prediction: Loads the trained model and vectorizer once at startup, applies the training-time pre-processing pipeline to new input, returns a Real/Fake label with a confidence score, and surfaces the top contributing features via the XAI layer.

Module 06: User Interface: A React front end that accepts pasted text, uploaded documents/images (OCR), or voice transcripts; performs client-side validation; and renders color-coded verdicts, confidence bars, and a highlighted-text explanation view.

Module 07: Database & Admin: Persists users, articles, predictions and audit logs in a relational schema; exposes an administrator dashboard for user management, model retraining, security-log review, and platform-wide analytics (verdict distribution, classifier benchmarking).

VII. RESULTS AND EVALUATION

A. Test Strategy

Testing followed a layered strategy — unit, integration, functional, system, white-box, black-box, and acceptance testing — so that errors originating at different levels of the NLP/ML pipeline (tokenizer bugs, train/inference pre-processing mismatches, or API-layer failures) could be caught at the appropriate stage rather than only after full system assembly. Fifteen functional test cases were designed across registration, authentication, news submission, prediction, history, dataset management, model training, reporting and session handling; all fifteen passed with no defects encountered.

TABLE II. Representative Functional Test Case Results

ID	Module	Scenario	Status
TC-05	News Submission	Submit valid news article	Pass
TC-06	News Submission	Submit empty content	Pass
TC-07	Fake News Detection	Predict genuine news article	Pass
TC-08	Fake News Detection	Predict fabricated news article	Pass
TC-09	Prediction Result	Display confidence score	Pass
TC-11	Dataset Upload	Upload valid CSV dataset	Pass
TC-13	Model Training	Train ML model on dataset	Pass

B. Classifier Benchmarking

Four classical scikit-learn classifiers — Logistic Regression, Passive-Aggressive, Naive Bayes and Random Forest — were cross-evaluated on a held-out test split within the platform's Analytics module. The Naive Bayes classifier, used as the illustrative benchmark, achieved an accuracy of 88%, an F1-score of 88.3%, and a precision of 85%, as summarized in Table III. Consistent with prior comparative studies [4][7], the deep-learning and transformer-based

models (LSTM, BERT) trade additional computational and

Metric	Value
Accuracy	88%
F1-Score	88.3%
Precision	85%
Classifiers Benchmarked	Logistic Regression, Passive-Aggressive, Naive Bayes, Random Forest

infrastructure cost for improved contextual accuracy, motivating the system's configurable, multi-model design rather than commitment to a single algorithm.

TABLE III. Sample Classifier Benchmark (Naive Bayes)

C. Operational Snapshot

In deployment, the platform's dashboard reports cumulative auditing activity — total articles scanned, the Real/Fake verdict split, and the currently active "champion" classifier — giving administrators a continuously updated view of platform-wide credibility distribution alongside the per-article confidence scores produced at prediction time.

VIII. DISCUSSION

A. Key Contributions

This work makes three primary contributions. First, a strictly order-preserved, leakage-free pre-processing and feature-extraction pipeline ensures that training and inference remain consistent, a common source of silent accuracy degradation in NLP systems. Second, the multi-classifier design — spanning classical ML, LSTM, and BERT — makes the accuracy–cost trade-off explicit and configurable rather than fixed at design time. Third, the coefficient-based explainability layer moves the system beyond a bare probability score toward a human-interpretable, sentence-level rationale, directly addressing a known trust gap in automated content-moderation tools.

B. Comparison with Existing Approaches

Against manual fact-checking, the system reduces verification latency from hours or days to seconds, at the cost of relying on textual and linguistic cues rather than deep investigative research. Against simple keyword or domain-blacklist filters used by social platforms, the learned linguistic representations generalize to previously unseen phrasing and paraphrased misinformation that static rules cannot recognize.

C. Limitations

The current system is English-only and text-only, and does not yet incorporate the accompanying images, videos, or propagation patterns that increasingly accompany real-world misinformation. Classifier weights and thresholds, while empirically reasonable, have not been validated against a large-scale, continuously updated ground-truth corpus, and the explainability layer currently applies only to linear models, not to the LSTM or BERT variants.

IX. CONCLUSION AND FUTURE SCOPE

A. Conclusion

This paper has presented an AI Based Fake News Detection system that combines a leakage-free NLP pre-processing pipeline, a configurable suite of classical and deep-learning classifiers, and a coefficient-based explainability layer within a full-stack, database-backed web application. Layered testing across fifteen functional test cases achieved a 100% pass rate, and classifier benchmarking confirms that lightweight classical models remain competitive while transformer-based models offer superior contextual accuracy — confirming that a practical, explainable, and freely deployable fake-news screening tool can materially narrow the gap between the speed of misinformation and the speed of verification.

B. Future Scope

Planned enhancements include: (1) multilingual and regional-dialect support; (2) multimodal analysis of accompanying images, video and metadata; (3) a browser extension for real-time, in-feed flagging; (4) extending the explainability layer to LSTM and BERT models via attention-visualization or SHAP/LIME; (5) a continuous, human-in-the-loop retraining pipeline; and (6) integration with social-media APIs for proactive, large-scale monitoring of trending content.

. ACKNOWLEDGMENT

The author thanks Mr. P. Sivakumar, Asst. Professor, Department of Computer Applications, for his continuous guidance and supervision throughout this project, and Dr. S. Menaka, Head of the Department, for her support. The author also acknowledges Mr. Ujjwal, Project Head, Maincrafts Technology, for external guidance during the internship period (14 December 2025 – 14 March 2026), and the faculty of the Department of Master of Computer Applications, Nehru Institute of Information Technology and Management, Coimbatore, Anna University, for their encouragement throughout this work.

. REFERENCES

- [1] T. Jadhav, S. Adlinge, N. Dande, and S. Jadhav, "Fake News Detection on Social Media Using NLP," *International Journal of Scientific Research in Science and Technology*, vol. 12, 2025.
- [2] "Fake News Detection System Using NLP and Machine Learning," *ResearchGate Publication*, 2025.
- [3] R. Oshikawa, J. Qian, and W. Y. Wang, "A Survey on Natural Language Processing for Fake News Detection," *Proceedings of LREC*, 2020.
- [4] "Fake News Detection Using Machine Learning and Deep Learning Algorithms: A Comprehensive Review and Future Perspectives," *MDPI Information*, 2025.
- [5] "Fake News Detection using Stance Classification: A Survey," *arXiv preprint, IT University of Copenhagen*.

- [6] "Fake News Detection using NLP," IEEE Conference Publication, IEEE Xplore, 2023.
- [7] "A Comprehensive Survey on Machine Learning Approaches for Fake News Detection," Multimedia Tools and Applications, Springer, 2024.
- [8] "A Survey on Role of Machine Learning and NLP in Fake News Detection on Social Media," IEEE Conference Publication, IEEE Xplore.
- [9] "A Survey on Fake News Detection using NLP," IJCRT, vol. 11, no. 2, 2023.
- [10] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake News Detection on Social Media: A Data Mining Perspective," ACM SIGKDD Explorations Newsletter, 2017.
- [11] X. Zhou and R. Zafarani, "A Survey of Fake News: Fundamental Theories, Detection Methods, and Opportunities," ACM Computing Surveys, 2020.
- [12] scikit-learn Documentation. [Online]. Available: <https://scikit-learn.org>
- [13] NLTK Documentation. [Online]. Available: <https://www.nltk.org>
- [14] Hugging Face Transformers Documentation. [Online]. Available: <https://huggingface.co/docs/transformers>
- [15] Kaggle Fake and Real News Dataset. [Online]. Available: <https://www.kaggle.com/datasets/clmentbisailon/fake-and-real-news-dataset>